

Business Data Analysis

MGM509

Lecture 9

Working with Unstructured Data: Text Mining and Representation

Prepared by
Eng. Ahmed Hussein, MBA, PfMP®,
PMP®, CMP APMG

Lecture
9

What we will cover

1. Dealing with Text Data
2. Text Preprocessing Techniques
3. Representing Text Numerically
4. Text Classification Techniques
5. Unsupervised Mining & Beyond

Lecture
09

Dealing with Text Data



Lecture 9

Why Text Data Matters?

Text Is Everywhere

- majority of modern data is **unstructured text** — from emails, chat logs, social media, and product reviews to documents and transcripts.

Hidden Insights

- Text contains **rich qualitative signals** that support tasks like classification (e.g., spam detection), prediction (e.g., churn), and personalization (e.g., product recommendations).

Broad Applications Across Sectors

- Used in **customer support** (ticket triaging), **finance** (risk disclosures), **healthcare** (clinical notes), and **marketing** (brand monitoring).

Analytical Challenge

- Unlike numerical data, text must be **converted into structured representations** (vectors) to apply statistical or machine learning models.

Competitive Advantage

- Organizations that successfully extract insights from text data gain a **strategic edge** in decision-making, automation, and customer engagement.



Unstructured Nature



Ambiguity & Context
Dependence



High Dimensionality



Informal Language
& Noise

Lecture
9

Challenges of Text Data

1. Unstructured Nature

- Text lacks formal structure — unlike numbers or tables — making it harder to process directly.

2. High Dimensionality

- Each unique word adds a feature. Even small datasets can have thousands of terms, leading to sparsity and overfitting.

3. Ambiguity and Context Dependence

- Words can have multiple meanings depending on context (e.g., “bank” as a financial institution vs. a riverbank).

4. Informal Language & Noise

- Misspellings, slang, emojis, abbreviations, and grammatical errors are common in user-generated content.

5. Lack of Labeling

- Most text data is unlabeled, making supervised learning tasks more challenging without manual annotation.

Lecture 9

Common Applications



Spam detection (classification)

- * Filter unwanted or malicious messages using word patterns and probabilistic classification.
- * Example: Gmail filtering spam using Naive Bayes models.



Topic modeling (unsupervised)

- * Uncover hidden themes or group similar documents together.
- * Example: Organizing articles or news content into topical categories.



Sentiment analysis

- * Understand customer opinions from reviews, social media, or surveys to guide product and marketing strategies.
- * Example: Classifying reviews as positive/negative on Amazon.

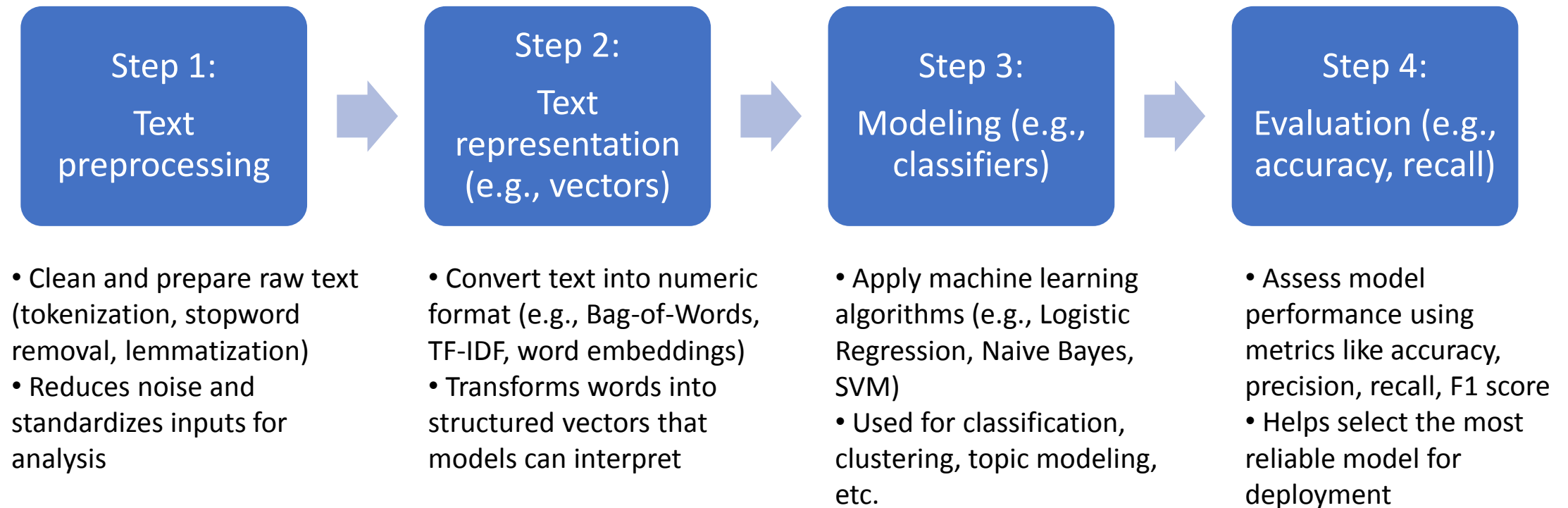


Customer Support Automation, Chatbots and recommendation engines

- * Classify and route support tickets, power chatbots, and mine FAQs using cleaned and structured text.
- * Example: Grouping similar customer issues for escalation.

Lecture 9

Overview of the Process



Lecture
09

Text Preprocessing Techniques



Lecture
9

Tokenization

- The process of splitting raw text into tokens — usually words, phrases, or subwords.
- Removes Punctuation and Symbols: Unnecessary characters like punctuation, emojis, or special symbols are often removed or separated.
- Tokenization is the first step before applying further NLP tasks like stemming, lemmatization, or vectorization.
- Example:

“Data science is powerful” → [‘Data’, ‘science’, ‘is’, ‘powerful’]

Lecture
9

Stopword Removal

- **Stopwords are Common words that appear frequently but add little informational value (e.g., “the”, “is”, “and”, “of”).**
- **Why Remove Them?**
 - They clutter the feature space and dilute the signal of more meaningful terms during modeling.
 - Reduces Dimensionality: By removing stopwords, we shrink the vocabulary size and reduce computational complexity.
 - Improves Model Focus and Accuracy: Fewer irrelevant terms = cleaner input = better generalization in classification and clustering tasks.
- **Example:**

[“Data”, “science”, “is”, “the”, “future”, “of”, “business”] → [“Data”, “science”, “future”, “business”]

Lecture 9

Stemming and Lemmatization

- Both techniques reduce words to a base form to group similar terms and reduce redundancy in text data.
- We use them to:
 - Reduces vocabulary size
 - Groups together variations of the same concept
 - Improves consistency for vectorization and modeling

Stemming	Lemmatization
• Cuts off word endings (heuristic-based) • Example: “running” → “run”, “happily” → “happi” • Fast but can produce non-real words	• Maps words to dictionary base form (lemma) using context • Example: “was” → “be”, “better” → “good” • More accurate but slower than stemming
Use stemming for speed	Use lemmatization for semantic accuracy

Lecture 9

Lowercasing and Cleaning - Normalizing Raw Text

Normalize for Consistency:

- Prepares text for tokenization and vectorization by reducing variation.
- Ensures “Email”, “email.”, and “EMAIL” are all interpreted the same.

Reduces Noise:

- Improves model efficiency by focusing only on meaningful textual content.

Lowercasing: Ensures all words are treated equally by converting everything to lowercase (e.g., “Data” → “data”).

Remove Digits and Symbols: Cleans out irrelevant characters such as numbers, punctuation, emojis, and HTML tags.

- Example: “Great product!!! 🤔 🤔 ” → “great product”

Lecture
09

Representing Text Numerically



Lecture
9

Bag of Words (BoW)

- Represents a document as a vector of word counts (frequency) — each unique word is a feature.
- Ignores Grammar and Word Order, Focuses only on how often a word appears, not where it appears or how it's used in context.
- Simple and effective baseline, a strong **baseline** approach for many classification tasks like spam detection, sentiment analysis.
- Limitations: Can lead to sparse matrices and lacks understanding of meaning or context.

Example:

- Text 1: “Data is powerful”,
- Text 2: “Powerful data tools”
- Vocabulary: [“data”, “is”, “powerful”, “tools”]

- BoW Vectors:

Text 1 → [1, 1, 1, 0]

Text 2 → [1, 0, 1, 1]

Lecture
9Vocabulary and Document-Term
Matrix

- **Vocabulary** = Unique Word Set
- **Document-Term Matrix (DTM)**: A structured matrix representation where:
 - Rows = documents
 - Columns = vocabulary terms
 - Cell values = word frequencies
- Use in ML: Forms the input feature matrix for classification, clustering, and other text mining tasks.

Document	data	science	tools
Doc 1	1	1	0
Doc 2	1	0	1

Lecture
9TF-IDF (Term Frequency-Inverse
Document Frequency)

- TF-IDF is a weighting scheme that balances a word's frequency in a document (TF) against its rarity in the overall corpus (IDF).
- **Term Frequency (TF):** Measures **how often** a word appears in a document.
- **Inverse Document Frequency (IDF):** Measures how unique a word is across all documents. Common words like "is" get low IDF, Rare words like "science" get high IDF
- **TF-IDF Score = $TF \times IDF$** - Emphasizes meaningful words and downweights common ones.
- **Example:** Document: "data science is fun"

TF: [1, 1, 1, 1]

IDF: high for "science", low for "is"

TF-IDF vector: higher values for "science" and "data", lower for "is"

Lecture
9

Pros and Cons of TF-IDF

Pros	Cons
Highlights important and unique words	Ignores word order and context
Simple, efficient, and easy to implement	No semantic understanding (e.g., synonyms not linked)
Strong baseline for classification and clustering	Produces sparse, high-dimensional feature vectors
Effective for document filtering and keyword extraction	Cannot distinguish between nuanced meanings of words

Lecture
9

Vector Space Model (VSM)

- A framework for representing text documents as vectors in a high-dimensional space, where each dimension corresponds to a unique term in the vocabulary.
- How It Works
 - Each document becomes a vector of term weights (e.g., word counts or TF-IDF scores).
 - Similar documents are close in vector space; dissimilar ones are far apart.
- Use Case: Information Retrieval: Search engines use VSM to find documents similar to a query by computing cosine similarity between vectors.
- Enables fast document comparison and clustering
- Foundation for algorithms like KNN, SVM, and vector-based search

Lecture
9

Cosine Similarity

- Cosine similarity calculates the cosine of the angle between two document vectors in high-dimensional space

$$\text{Cosine Similarity} = \frac{A \cdot B}{\|A\| \times \|B\|}$$

- It ignores magnitude and focuses on the direction, making it ideal for comparing documents of varying lengths.
- **Range:** 0 (no similarity) to 1 (identical)
- Used in search engines to find documents most similar to a query, and Recommender systems to suggest similar items or content

Lecture 9

Example:

- Two Documents:
 - Doc A: “Data science is powerful”
 - Doc B: “Science tools are powerful”
- **Vocabulary:** [data, science, is, powerful, tools, are]
- **Cosine Similarity:** You can now compute similarity between Doc A and Doc B by measuring the cosine of the angle between the vectors.
- This allows the model to retrieve, cluster, or classify documents based on content similarity.

Document Vectors (Bag of Words):

Word	Doc A	Doc B
data	1	0
science	1	1
is	1	0
powerful	1	1
tools	0	1
are	0	1

Lecture
9

Limitations of Basic Representations

No Word Order Awareness

- Bag-of-Words and TF-IDF treat text as a “bag” — ignoring the sequence of words.
- Example: “not good” and “good” are treated similarly.

Cannot Understand Word Meaning

- These models don’t grasp the semantics of words — they treat every term as independent.
- Example: “happy” and “joyful” are unrelated in vector space.

Polysemy Confusion

- A single word with multiple meanings (e.g., “bank”) is represented as one feature, losing context.

No Contextual Relationships

- Words that often appear together or rely on neighboring words for meaning are not linked.

Motivated the Rise of Semantic Models

- These limitations led to the development of word embeddings (Word2Vec, GloVe) and contextual models (BERT) that understand meaning and relationships.

Lecture 9

Transition to Mining Text

Representation Enables Analysis

- Once text is transformed into vectors (e.g., TF-IDF or embeddings), it becomes structured and suitable for analysis.

Apply Machine Learning Techniques

- Use standard algorithms to extract insights: Classification (Spam detection, sentiment analysis), Clustering (Group similar documents), or Association (Discover co-occurring terms or patterns)

Text Becomes Structured Data

- Documents, emails, reviews, and chats can now be analyzed like numerical data in a spreadsheet.

Enables Automation

- Helps automate tagging, categorization, recommendation, and more across industries.

Leads to Powerful Applications

- From chatbots and search engines to fraud detection and voice assistants.

Lecture
09

Text Classification Techniques



Lecture
9

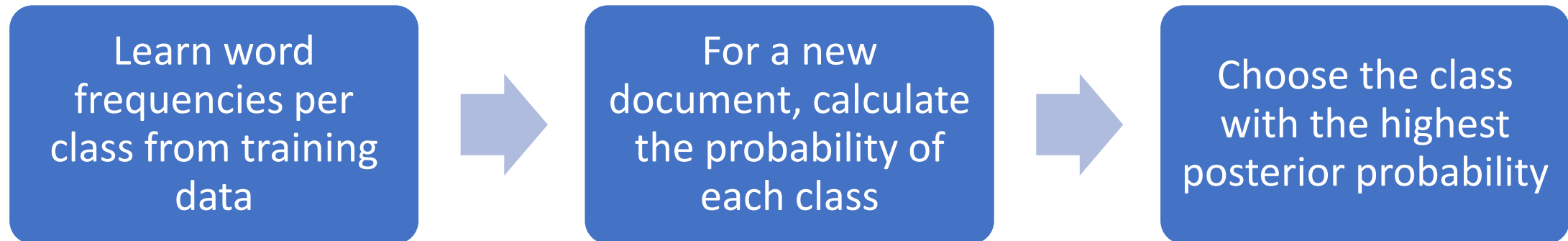
Introduction to Text Classification

- Assigning predefined categories or labels to text documents based on their content.
- Requires labeled training data where each text sample is already assigned a class (Supervised Learning).
- Common Examples:
 - Spam Detection: Classify emails as spam or not spam
 - Sentiment Analysis: Label reviews as positive, neutral, or negative
 - Topic Categorization: Assign news articles to topics like sports, politics, tech
- Algorithms → Logistic Regression, Naive Bayes, Support Vector Machines, and Neural Networks.
- Applications: Customer feedback analysis, content moderation, email filtering, and helpdesk automation.

Lecture
9

Naive Bayes for Text

- Assumes that words appear independently of each other — even though this isn't true in real language, the model still works well in practice.
- How It Works:



- Example: Classify a review as positive or negative based on the likelihood of words like “great”, “bad”, “terrible”, “excellent” occurring in each class.

Lecture
9

Logistic Regression

- A linear model that estimates the probability that a document belongs to a certain class, based on its word vector
- Computes a weighted sum of input features (words), then applies the logistic (sigmoid) function to get a probability between 0 and 1.

$$P(y = 1|x) = \frac{1}{1 + e^{-(w \cdot x + b)}}$$

- Easy to Train and Tune, each weight shows how strongly a word contributes to the classification — helpful for feature understanding

Lecture
9

SVMs and Other Methods



Support Vector
Machines: finds
boundary between
classes



Decision Trees,
Random Forests
also applicable



Model choice
depends on data
and accuracy needs

Lecture
9

Feature Selection for Text

- Text data can contain thousands of unique words (features), many of which are irrelevant or noisy.
- Feature Selection --> Reduces training time and model complexity, Improves generalization, and Enhances interpretability of models
- Common Techniques:
 - Chi-Square Test: Measures the independence between a word and a class
 - Mutual Information: Quantifies the shared information between word and label
 - Frequency Thresholds: Removes rare or overly frequent terms (e.g., words that appear in >90% of documents)
- Use before applying models like Naive Bayes, SVM, or Logistic Regression.

Lecture
9

Model Evaluation

- Use classification metrics to assess model performance:
 - Accuracy: Overall correctness
 - Precision: Correct positives among predicted positives
 - Recall: Correct positives among actual positives
 - F1-Score: Harmonic mean of precision & recall (balances both)
- Validation Strategies:
 - Holdout Set: Split data into training and test sets (e.g., 80/20)
 - Cross-Validation: Rotate validation sets across folds for more reliable estimates

Lecture
09

Unsupervised Mining & Beyond



Lecture
9

Clustering Text Documents

Clustering groups similar documents together based on their content — without needing predefined labels. It's an unsupervised learning technique.

Common Algorithms: K-Means Clustering: Groups documents into k clusters using vector similarity and Hierarchical Clustering: Builds a tree of document similarity (dendrogram)

Key Applications:

Topic Discovery: Group news articles or research papers by topic

Document Organization: Sort emails, customer reviews, or files automatically

User Segmentation: Group users by similar comment or behavior patterns

How It Works:

Convert documents into
vectors (TF-IDF or
embeddings)



Use distance metrics (e.g.,
cosine similarity) to measure
document closeness



Assign clusters based on
proximity

Lecture
9

Latent Dirichlet Allocation (LDA)

- LDA is a probabilistic model used to discover hidden topics in a collection of documents. It assumes each document is a mixture of topics, and each topic is a distribution over words.
- How It Works (Simplified):
 - Documents are made of topics
 - Topics are made of words
 - LDA reverse-engineers the document collection to uncover the most likely topic-word combinations
- What It Produces:
 - Topic-word distributions (e.g., Topic 1: “delivery”, “shipping”, “late”)
 - Document-topic distributions (e.g., Doc 12 is 60% Topic 1, 40% Topic 3)

Lecture
9

Association Rule Mining in Text

- Association rule mining uncovers frequent patterns or word co-occurrences within a text corpus.
- Treats each document as a “transaction” and each word or phrase as an “item”
- Use metrics like:
 - Support: How often items co-occur
 - Confidence: Likelihood of one word appearing if another does
 - Lift: How much more likely words are to appear together than by chance

Lecture
9

Association Rule Mining in Text

- Example: If many reviews contain both “slow delivery” and “poor packaging”:
 - Rule: {slow delivery} $\rightarrow$ {poor packaging}
 - Confidence: 75%, Lift: 1.8
- Applications:
 - Phrase extraction from reviews or complaints
 - Discovering topic co-occurrence patterns
 - Enhancing search engine suggestions or chatbot intent recognition

Lecture
9

Word Embeddings (e.g., Word2Vec)

- Word embeddings represent words as dense numerical vectors in a continuous space, capturing semantic relationships and contextual meaning.
- Example: Word2Vec Learns vector representations based on surrounding words (context window)
- Words used in similar contexts are placed close together in vector space
- **Benefits:**
 - Captures similarity (e.g., “king” $\approx$ “queen”)
 - Supports analogy reasoning: $\text{vector}(\text{"king"}) - \text{vector}(\text{"man"}) + \text{vector}(\text{"woman"}) \approx \text{vector}(\text{"queen"})$
 - Preserves semantic relationships better than Bag of Words or TF-IDF

**Lecture
9**

Word Embeddings (e.g., Word2Vec)

Advantages Over BoW:

- Smaller and denser vectors
- Includes contextual nuance
- Suitable for advanced models like deep learning, recommendation systems, and text generation

Applications:

- Semantic search
- Text classification & clustering
- Machine translation
- Chatbots and virtual assistants

Limitations, Challenges, and Ethical Considerations

Limitations and Challenges:

- Biased or toxic language in training data
- Interpretability of complex models
- Need for domain-specific preprocessing
- Privacy concerns with text data

Ethical Considerations:

- Bias propagation and fairness
- Transparency in decision-making systems
- Responsible Deployment

Lecture
9

Trends in Text Mining

Rise of Pretrained Language Models

- Models like BERT and GPT dominate NLP tasks
- Enable transfer learning and state-of-the-art performance

Contextual Understanding with Embeddings

- Move beyond static word vectors to context-aware models
- Improves semantic accuracy and disambiguation

Explainable NLP (XAI for Text)

- Demand for interpretable models in sensitive applications
- Tools like LIME and attention maps aid transparency

Multilingual & Cross-Lingual Capabilities

- Models now handle multiple languages in one system
- Crucial for global businesses and inclusive AI

Privacy & Ethics in Text Processing

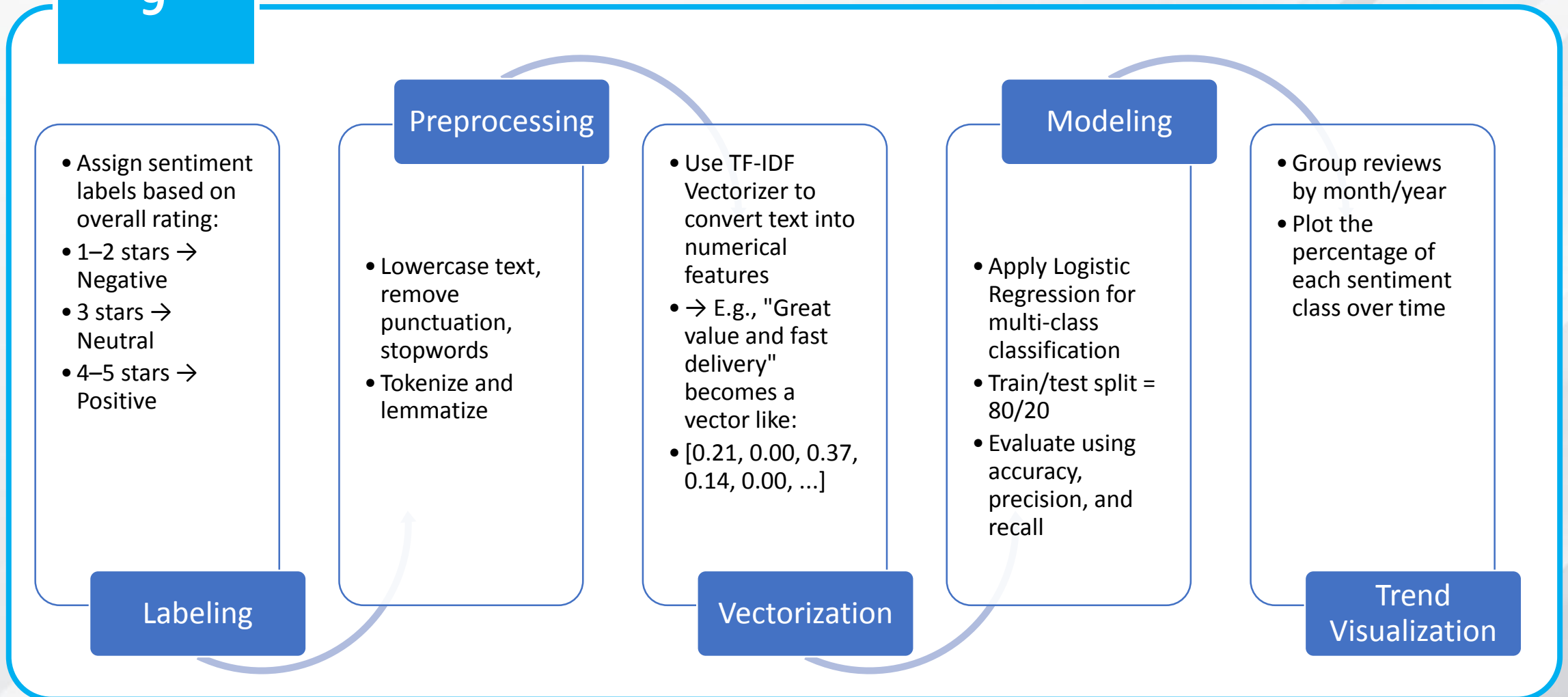
- Focus on data protection, bias mitigation, and fairness
- Privacy-preserving methods like federated learning gain traction

Lecture
9Case Study – Amazon Product
Reviews

- Source: Amazon Reviews Dataset (can also use Yelp or IMDB datasets)
- Sample Size: 10,000 reviews
- Columns:
 - reviewText: the raw customer review
 - overall: numeric rating (1–5 stars)
 - reviewTime: date of review

Lecture 9

Case Study – Amazon Product Reviews

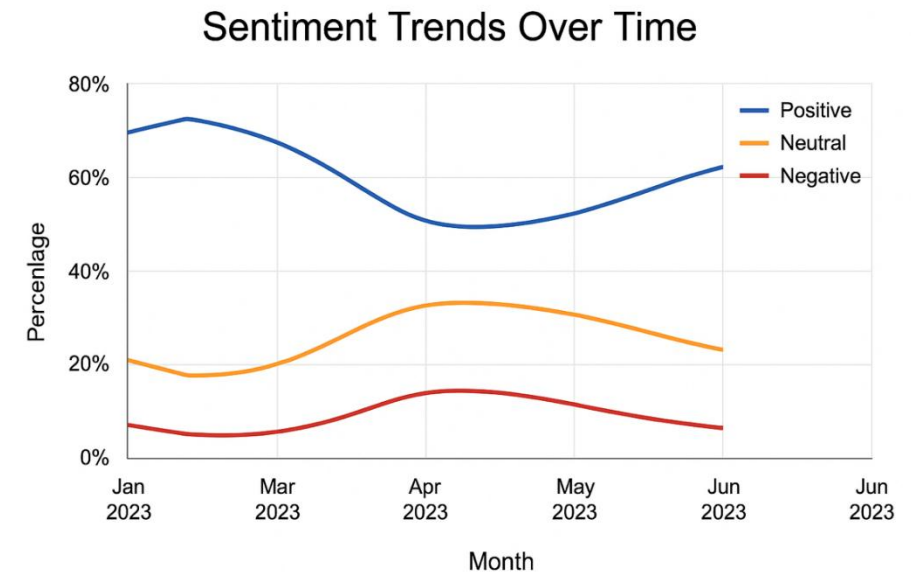


Lecture
9

Case Study – Amazon Product Reviews

- Example Output:

Month	Positive	Neutral	Negative
Jan 2023	68%	18%	14%
Feb 2023	74%	15%	11%
Mar 2023	65%	20%	15%



Lecture 9

Key Points

•Text Must Be Structured to Be Analyzed

- Raw text is unstructured; we must convert it into numerical representations (e.g., Bag of Words, TF-IDF, embeddings) before applying
- models.

\.Preprocessing Is Foundational

- Steps like tokenization, stopword removal, stemming/lemmatization, and cleaning are critical for reducing noise and improving model
- performance.

\.Representation Techniques Affect Results

- Simple methods (BoW, TF-IDF) are interpretable but ignore context; word embeddings and contextual models (like Word2Vec, BERT)
- capture meaning more effectively.

\.Text Mining Enables Diverse Applications

- Techniques like classification, clustering, topic modeling (LDA), and sentiment analysis uncover insights across domains—from
- product reviews to fraud detection.

\.Ethics and Interpretability Matter

- Models must be evaluated for bias, transparency, and fairness—especially when dealing with user-generated content or sensitive
- domains.

For Discussion Panel and Next Lecture:

- Chapter 10 “Representing and Mining Text”
- Download Amazon Reviews Dataset and practice developing a sentiment analysis based on customer reviews

**Lecture
9**



**For Discussion Panel
and Next Lecture**



Thank You